Computação 2

Aula 6B

Vetor de Estruturas (*structs*)

Profª. Fabiany
fabianyl@utfpr.edu.br

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Vetor de estruturas (structs)

- Podemos declarar um vetor de estruturas, para isso primeiro definimos um tipo estrutura e então declaramos uma variável vetor desse tipo.

Struct - Exemplo 1

```
#include <stdio.h>

void main()
{

 // Queremos cadastrar 4 pessoas diferentes
  char nome1[50], nome2[50], nome3[50], nome4[50];

  int idade1,idade2,idade3, idade4;

  char rua1[50], rua2[50], rua3[50], rua4[50];

  int  numero1, numero2, numero3, numero4;


  ....



}
```

Vetor de Structs - Exemplo 1

```
#include <stdio.h>

struct  pessoa{
     char nome[50], rua[50];
      int idade, numero;
};

void main()
{

 // Queremos cadastrar 4 pessoas diferentes
  struct pessoa p1, p2, p3, p4;

}
```

```
#include <stdio.h>

struct  pessoa{
     char nome[50], rua[50];
      int idade, numero;
};

void main()
{

 // Queremos cadastrar 4 pessoas diferentes
 //Vetor de struct
 struct pessoa p[4];

}
```

Vetor de structs - Exemplo 1

```
#include <stdio.h>

struct  pessoa{
     char nome[50], rua[50];
      int idade, numero;
};

void main()
{

 // Queremos cadastrar 4 pessoas diferentes
 //Vetor de struct
 struct pessoa p[4];

//trabalhando com os elementos do vetor de structs
p[0].idade = 31;

strcpy(p[1].nome, " Ricardo");

p[2].numero  p[0].numero – 1;

}
```

Vetor de structs - Exemplo 2

```
#include <stdio.h>
  struct   ficha{
        char nome[30];
        float media;
    };

   void main() {
      struct ficha aluno[10];
      int i;
      for (i = 0; i < 10; i++) {
            printf("Digite o nome do aluno\n");
            scanf("%s", aluno[i].nome);                  //utilizar os índices para acessar a posição desejada
            printf("Digite a media do aluno\n");         //igual vetor e matriz que conhecemos
            scanf ("%i", &aluno[i].media);
            printf("O aluno %s tem media igual a %f \n ", aluno[i].nome, aluno[i].media);
       }
  }
```

Passando estruturas para funções

- As estruturas podem ser passados como parâmetros de uma função, como qualquer outro tipo.
- A estrutura deve ser declarado antes da função.
- O parâmetro formal recebe uma cópia da estrutura, da mesma forma que em uma atribuição envolvendo estruturas.
- Podemos passar por parâmetro apenas um campo (elemento) da estrutura para uma função como uma variável simples.
- Podemos passar uma estrutura inteira como parâmetro para uma função.
- Podemos passar um vetor de estruturas como parâmetro para uma função.

Struct - Exemplo 1

```
int Calc_Idade (int a_pessoa, int a_atual )
{
    int id;

    id = a_atual – a_pessoa;

    return id;
}
```

```
#include <stdio.h>

struct Pessoa
{
    int dia;
    int mes;
    int ano;
    int idade;
};

int main()
{
   struct Pessoa Einstein, Newton;

   Einstein.dia    = 14;
   Einstein.mes  = 3;
   Einstein.ano  = 1879;

   Newton.dia    = 4;
   Newton.mes  = 1;
   Newton.ano   = 1643;

   Einstein.idade = Calc_Idade ( Einstein.ano, 2014 );
   Newton.idade  = Calc_Idade ( Newton.ano, 2014);

   printf ( "A idade de Einstein seria %d \n", Einstein.idade );
   printf ( "A idade de Newton seria %d  \n", Newton.idade );

  return 0;
}
```

Struct - Exemplo 2

```
int Calc_Idade (struct pessoa p, int a_atual )
{
    int id;

    id = a_atual – p.ano;

    return id;
}
```

```
#include <stdio.h>

struct Pessoa
{
    int dia;
    int mes;
    int ano;
    int idade;
};

int main()
{
   struct Pessoa Einstein, Newton;

   Einstein.dia    = 14;
   Einstein.mes  = 3;
   Einstein.ano  = 1879;

   Newton.dia    = 4;
   Newton.mes  = 1;
   Newton.ano   = 1643;

   Einstein.idade = Calc_Idade ( Einstein, 2014 );
   Newton.idade  = Calc_Idade ( Newton, 2014);

   printf ( "A idade de Einstein seria %d \n", Einstein.idade );
   printf ( "A idade de Newton seria %d  \n", Newton.idade );

  return 0;
}
```

Struct - Exemplo 3

```
int Calc_Idade (struct Pessoa p, int d, int m, int a)
{
   int idade = a - p.ano;
   if ( p.mes > m )
   {
      idade = idade - 1;
   }
   else
   {
        if (p.mes == m)
        {   if (p.dia > d)
            {
               idade = idade - 1;
            }
        }
   }
   return idade;
}
```

```
#include <stdio.h>
#include <string.h>
struct Pessoa
{
    char nome[30];
    int dia;
    int mes;
    int ano;
    int idade;
};

int main() {
   struct Pessoa Lista_Genios[2];

   strcpy ( Lista_Genios[0].nome, “Albert Einstein” );
   Lista_Genios[0].dia    = 14;
   Lista_Genios[0].mes  = 3;
   Lista_Genios[0].ano  = 1879;

   strcpy ( Lista_Genios[1].nome, “Isaac Newton” );
   Lista_Genios[1].dia    = 4;
   Lista_Genios[1].mes  = 1;
   Lista_Genios[1].ano   = 1643;

   Lista_Genios[0].idade = Calc_Idade (Lista_Genios[0], 13,05,2014);
   Lista_Genios[1].idade  = Calc_Idade (Lista_Genios[1],13,05,2014);


   printf("A idade de %s seria %d \n",  Lista_Genios[0].nome,
                                        Lista_Genios[0].idade );

   printf("A idade de %s seria %d  \n", Lista_Genios[1]..nome,
                                        Lista_Genios[1].idade );

return 0;
  }
```

Exemplo 4 - Struct

```
#include <stdio.h>
#include <string.h>

struct Pessoa
{
    char nome[30];
    int dia;
    int mes;
    int ano;
    int idade;
};
```

p é um parâmetro por valor aqui...

```
int Calc_Idade (struct Pessoa p, int dia, int mes, int ano)
{
    int idade = ano - p.ano;
    if ( p.mes > mes )
    {
        idade = idade - 1;
    }
    else
    {
        if (p.mes == mes)
        {    if (p.dia > dia)
            {
                idade = idade - 1;
            }
        }
    }
    return idade;
}
```

```
int main()
{
   struct Pessoa Lista_Genios[2]

   strcpy ( Lista_Genios[0].nome, "Albert Einstein" );
   Lista_Genios[0].dia    = 14;
   Lista_Genios[0].mes  = 3;
   Lista_Genios[0].ano  = 1879;

   strcpy ( Lista_Genios[1].nome, "Isaac Newton" );
   Lista_Genios[1].dia    = 4;
   Lista_Genios[1].mes  = 1;
   Lista_Genios[1].ano   = 1643;

   Lista_Genios[0].idade = Calc_Idade (Lista_Genios[0], 8, 2, 2007);
   Lista_Genios[1].idade  = Calc_Idade (Lista_Genios[1], 8, 2, 2007);


   printf("A idade de %s seria %d \n",  Lista_Genios[0].nome
                                        Lista_Genios[0].idade );

   printf("A idade de %s seria %d  \n", Lista_Genios[1]..nome
                                        Lista_Genios[1].idade );


   getchar();
   return 0;
}
```

Exemplo 3 - Struct

```c
#include <stdio.h>

struct Pessoa
{
   char nome[30];
   int dia;
   int mes;
   int ano;
   int idade;
};
```

p é um parâmetro por referência aqui...

```c
void Calc_Idade (struct Pessoa *p, int dia, int mes, int ano)
{
   (*p).idade = ano - (*p).ano;

   if ( (*p).mes > mes )
   {
      (*p).idade = (*p).idade - 1;
   }
   else
   {
        if ( (*p).mes == mes )
        {   if ( (*p).dia > dia )
            {
                (*p).idade = (*p).idade - 1;
            }
        }
   }
}
```

```c
int main()
{
   struct Pessoa Einstein, Newton;

   strcpy ( Einstein.nome, "Albert Einstein" );
   Einstein.dia   = 14;
   Einstein.mes  = 3;
   Einstein.ano  = 1879;

   strcpy ( Newton.nome, "Newton");
   Newton.dia    = 4;
   Newton.mes  = 1;
   Newton.ano   = 1643;

   Calc_Idade ( &Einstein, 28, 9, 2007 );
   Calc_Idade ( &Newton, 28, 9, 2007 );


    printf("A idade de %s seria %d \n",  Einstein.nome
                                         Einstein.idade );
    printf("A idade de %s seria %d  \n",  Newton.nome
                                         Newton.idade );


   getchar();
   return 0;
}
```

Usa-se a forma *(*p).variável*

Exemplo 4 - Struct

```c
#include <stdio.h>

struct Pessoa
{
   char nome[30];
   int dia;
   int mes;
   int ano;
   int idade;
};
```

p é um parâmetro por referência aqui...

```c
void Calc_Idade (struct Pessoa *p, int dia, int mes, int ano)
{
   p->idade = ano – p->ano;

   if ( p->mes > mes )
   {
      p->idade = p->idade - 1;
   }
   else
   {
      if ( p->mes == mes )
      {   if ( p->dia > dia )
          {
             p->idade = p->idade - 1;
          }
      }
   }
}
```

```c
int main()
{
   struct Pessoa Einstein, Newton;

   strcpy ( Einstein.nome, "Albert Einstein" );
   Einstein.dia   = 14;
   Einstein.mes  = 3;
   Einstein.ano  = 1879;

   strcpy ( Newton.nome, "Newton");
   Newton.dia    = 4;
   Newton.mes  = 1;
   Newton.ano   = 1643;

   Calc_Idade ( &Einstein, 28, 9, 2007 );
   Calc_Idade ( &Newton, 28, 9, 2007 );

   printf ("A idade de %s seria %d \n",    Einstein.nome
                                           Einstein.idade );
   printf ("A idade de %s seria %d  \n",  Newton.nome
                                           Newton.idade );
   getchar();
   return 0;
}
```

Em vez da forma *(*p).variável* , usa-se mais a forma *p->variável.*

Exemplo 5 - Struct

```c
#include <stdio.h>
#include <stdlib.h>

struct Pessoa
{
    int idade;
    char nome [ 100 ];
    char sexo;
};

int quantidmaior (struct Pessoa fs[], int tam, float med)
{
    int quant = 0;

    int cont = 0;

    for ( cont = 0; cont < tam; cont = cont + 1 )
    {
       if ( fs[cont].idade > med)
       {
          quant = quant + 1;
       }
    }

    return quant;
}
```

```c
int main()
{
   struct Pessoa Funcs[3];

   int quant, contaid = 0;
   float media;

   printf ( "Programa dos dados de funcionários. \n \n \n" );

   int cont;
   for ( cont = 0; cont < 3; cont = cont +1 )
   {
      printf ("Digite o nome do funcionario número %i: \n", cont+1 );
      fflush ( stdin );
      gets ( Funcs[cont].nome );

      printf ( "Digite a idade do funcionario número %i: \n", cont+1);
      fflush ( stdin );
      scanf ( "%i", &Funcs[cont].idade );

      printf ( "Digite o sexo (f/m) do funcionario número %i: \n", cont+1 );
      fflush ( stdin );
      scanf ( "%c", &Funcs[cont].sexo);
      contaid = contaid + Funcs[cont].idade;

      printf ("\n");
   }

   media = (float) contaid/cont;

   quant = quantidmaior (Funcs, cont, media);

   printf ( " A média de idade é %.2f: \n", media );
   printf ( " A quantidade de funcionário com idada acima da média é %i: \n", quant );

   printf ("\n \n");
   system ("Pause");
   return 0;
}
```

Exercícios

1) Fazer um programa de ‘dialogo de login’ onde seja possível cadastrar no máximo 10 nomes de usuário e suas respectivas senhas (nomes de usuário repetidos devem ser descartados). No diálogo de login, o programa deve testar se o usuário fornecido existe e se a sua senha confere

Exercícios

2) Faça um programa que realize o cadastro de contas bancárias com as seguintes informações: número da conta, nome do cliente e saldo. O banco permitirá o cadastramento de apenas quinze contas e não poderá haver mais que uma conta com o mesmo número. Crie o menu de opções a seguir:

   1) Cadastrar conta;
   2) Fazer movimentações na conta (saque e deposito);
   3) Buscar cliente por conta e mostrar as informações;
   4) Excluir uma conta;
   5) Sair.

Exercícios

3) Faça um programa que efetue reservas de passagens áreas de determinada companhia. O programa deverá ler os números dos aviões e o numero de lugares disponíveis em cada avião. O programa deverá mostrar o seguinte menu de opções:

   1) Cadastrar os números dos aviões;
   2) Cadastrar o número de lugares disponíveis em cada avião;
   3) Reservar passagem (escolher o avião e o lugar);
   4) Consultar por avião;
   5) Sair.